Sketch-Based Navigation in 3D Virtual Environments

Benjamin Hagedorn and Jürgen Döllner

Hasso-Plattner-Institut at University Potsdam,
Prof.-Dr.-Helmert-Str. 2-3, 14482 Potsdam, Germany
benjamin.hagedorn|doellner@hpi.uni-potsdam.de

Abstract. Navigation represents the fundamental interaction technique in 3D virtual environments (3D VEs) as it enables the users to explore the 3D world and to interact with its objects. Efficient navigation strategies and techniques are required, which take account of the users and their goals and avoid problems of general navigation methods, such as "getting-lost" situations and confusing view configurations. This paper presents a novel method for specifying and controlling navigation in 3D VEs based on sketching navigation commands. The users sketch their navigation intentions on top of the perspective projection of the 3D scene. The system interprets these sketches regarding their geometry, spatial context, and temporal context. Unlike other sketchy navigation techniques, our approach identifies the hit objects of the underlying 3D scene and takes advantage of their semantics and inherent navigation affordances. The approach has been prototypically implemented for the exploration of a virtual 3D city model with a touch-sensitive display.

1 Introduction

Navigation, which is often referred to as "the aggregate task of wayfinding and motion" [4], denotes the fundamental interaction technique in 3D geovirtual environments (3D VEs) but still represents a non-trivial task for the user interface technology [7]. General navigation techniques (e.g., world-in-hand controls, fly-over controls, and virtual trackballs) give the users direct control over the navigation process. Common problems of these approaches include "getting-lost" situations, confusing view configurations, abrupt camera motion, and loss of visual contact to landmarks. Thus, to improve the usability of 3D interaction processes, navigation techniques have to assist users to navigate through and interact with the 3D world and its objects.

This paper presents a sketch-based navigation technique for 3D VEs such as virtual 3D city models and 3D landscape models. In our approach, the users express their navigation intentions in terms of graphical sketches by drawing curves and points to indicate paths and locations as well as pen-based gestures. These sketches are interpreted according to a pre-defined graphical vocabulary of navigation commands, taking into account their spatial and temporal context as well as the navigation affordances inherent to the elements of the 3D VE.
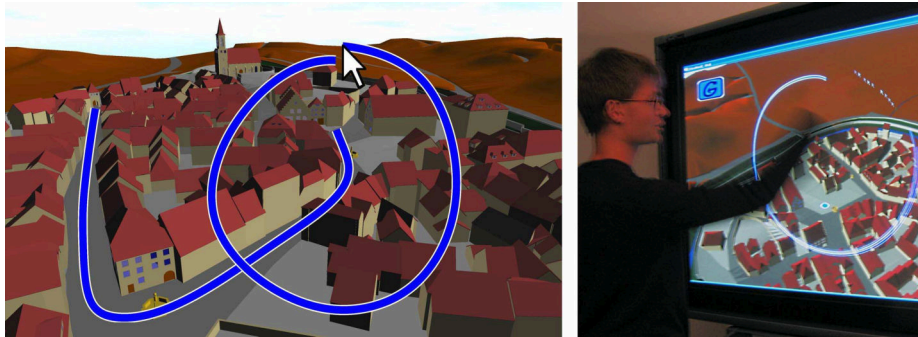
Fig. 1. Left: Example of a sketch-based navigation command applied to a complex virtual 3D city model. The user draws a curve on the street and combines this command with a circle like gesture. The derived animation will move the camera along the sketched path and finally rotate for inspecting the target area. – Right: Using the sketch-based navigation with a touch-sensitive smart board.

Sketch-based navigation is conceptually a higher-level navigation technique, as it relieves the users from controlling the motion task and even can assist in wayfinding. It can be used with any device allowing for 2D sketch input, e.g., mouse, graphic tablet, or touch-screen – see Fig. 1.

2   Related Work

Navigation constraints cope with the large number of degrees of freedom inherent to 3D navigation and represent a major technique in the field of assisted 3D navigation. Hanson and Wernert [9] propose designer-supplied constraints on the bases of 2D controllers, Tan et al. [13] introduce the speed-coupled flying, and Buchholz et al. [1] apply several navigation strategies for reaching a high orientation value. A detailed description of constraints for navigation in geovirtual environments can be found at Döllner [5]. StyleCam by Burtnyk et al. [2] represents an authoring-based approach constraining the camera movements in space and time. HoverCam by Khan et al. [12] assists users in panning, zooming and tumbling the virtual camera, particularly for single object inspection. Different from those, Russo dos Santos et al. [7] propose a navigation concept, which provides specific navigation methods according to the type of a 3D VE.

Sketching is a natural and intuitive input technique, which can be applied for 2D and 3D interaction. In the context of virtual 3D objects and virtual 3D worlds, sketching is often regarded as an effective means for modeling and manipulation [3][11][14]. Only few approaches seem to target at sketch-based navigation in 3D VEs. Igarashi [10] et al. introduce the concept of path drawing for 3D walkthroughs, which allows the users to sketch the path a virtual avatar shall move along on top of the perspective view of the 3D scene. Cohen et al. [3] suggest a method for sketching 3D curves and mention the possibility

to use them as a camera path and Hachet et al. [8] propose a technique, which uses a circle-shaped gesture for focus definition and a special widget for positioning the camera. Döllner et al. [6] present an approach for semantics-based navigation in 3D city models on mobile devices, which bases on sketching navigation commands by stylus and incorporates the type and meaning of the 3D city objects for deriving an appropriate navigation animation. The article at hand gives a detailed description of an extended sketch-based navigation concept and implementation.

3 Sketch-Based Navigation

3.1 Sketchy Navigation Commands

As described in [6], we conceptually distinguish two types of navigation sketches:

- *Object-related sketches* are associated with objects of the 3D VE and enable a semantics-based interpretation as the system can derive the intended navigation from the semantics of the marked scene objects.
- *Pen-based gestures* implement a complementary set of commands that are not bound to a spatial context.

Sketchy navigation commands allow for the following types of navigation interactions, which differ in their degree of navigation abstraction:

- *Camera-oriented navigation.* Users perform wayfinding by themselves. They think in terms of moving the camera "left and right", "up and down", etc.
- *Motion-oriented navigation.* Users sketch by drawings, where the camera shall move along and where to gaze. This allows for the definition of more complex navigation patterns, such as "drive along this path and look at that building".
- *Task-oriented navigation.* Users sketch complete tasks or subtasks to be fulfilled, e.g., "get an overview" or "inspect the building". Particularly, they no longer deal with camera positions and orientations or camera paths and gaze directions. Of course, task-oriented navigation commands heavily depend on the users and the user tasks.

Table 1. Examples of pen-based gestures. The black dot indicates a gesture's starting point.
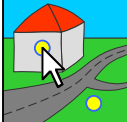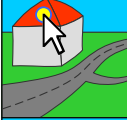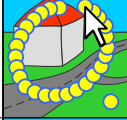
Sketch	Navigation Command	Sketch	Navigation Command	Sketch	Navigation Command
	Tilt up the camera.		Rotate the camera to right.		Take an overview position.
	Tilt down the camera.		Rotate the camera to left.		Undo last navigation.

Table 2. Examples of sketch-based navigation commands. Gestures can be used as modifiers for object-related sketches.

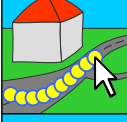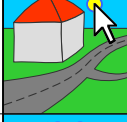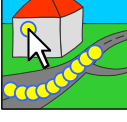| Sketch	Navigation Command & Result	Sketch	Navigation Command & Result
	Point on a building: Finding the shortest path to the building, driving there, and looking at the building.		**Point on the ground and point on a building:** Flying to the marked ground point and looking at the building finally.
	Point on a building's roof: Flying up to the roof, placing the camera on top and looking around.		**Point on the ground and circle-shaped gesture:** Flying to the marked ground point and looking around.
	Curve on a street: Driving along the street.		**Point on the sky:** Soaring above ground for overview.
	Curve on a street and point on a building: Driving along the street and looking at the building finally.		**Circle-shaped gesture and point on a building:** Soaring above ground and looking at the building finally.

In our approach, motion-oriented navigation commands are mainly defined by object-related sketches. Pen-based gestures cover user interface management operations (undo), low-level camera-oriented navigation commands (e.g., rotating or tilting), modify preceding object-related sketches, or trigger more complex navigations. Table 1 and Table 2 illustrate examples of sketchy navigation commands.

3.2 Navigation Affordances of 3D Objects

The navigation abilities and affordances of typical elements of 3D VEs play a key role in our approach. Taking into account the semantics of involved scene objects facilitates navigation strategies that can be adapted to specific users and tasks. These elements provide motion-oriented and task-oriented navigation affordances, as well as additional information that can be facilitated for the generation of appealing camera animations.

Thus, the sketch-based navigation technique requires for an appropriate model that not only contains geometry but also provides thematic information about the contained entities. Our implementation basis on the CityGML model, which is a specification for virtual 3D city and landscape models and supports, e.g., terrain models, vegetation models and detailed building models. The prototype implementation regards terrain, vegetation area, building, roof, street, and the sky as relevant object types.
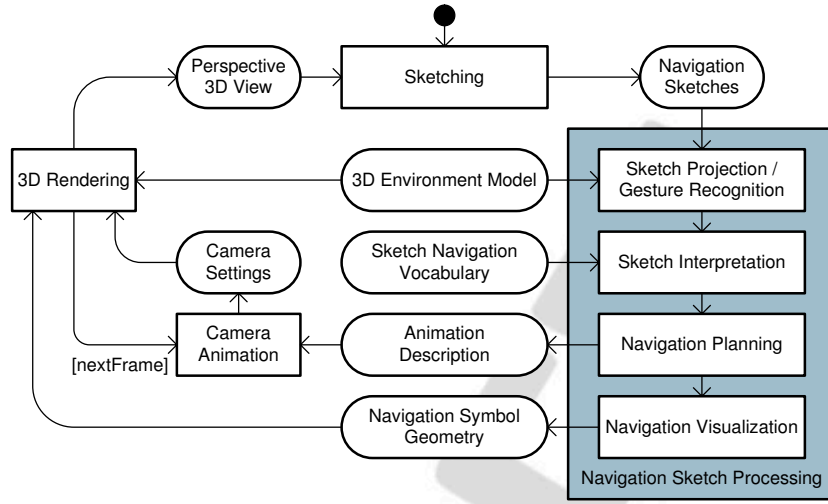
Fig. 2. Principal components of the sketch-based navigation command system.

4 Processing Sketch-Based Navigation Commands

The users enter object-related sketches by the left mouse button or by finger or stylus on touch-sensitive displays. For gesture input, the right mouse button or a view plane button are used. Additionally, sketch-based navigation processing comprises sketch recognition, sketch interpretation, and camera animation – see Fig. 2.

4.1 Interpreting Navigation Sketches

For the interpretation of navigation sketches, our technique takes into account the *sketch geometry* (curves and points), the *spatial context* (the virtual location to which the sketch is aligned or associated), and the *temporal context* (the sketch composition, command history, and drawing speed). [6]

Object-related sketches get evaluated from the scene graph by projecting each 2D sketch point (curves are discretized), calculating a set of ray intersections with the relevant objects, and retrieving the object types. For each sketch, its semantics is determined from the major type of the nearest to the camera intersection of each sketch point. All intersections corresponding to this type represent the projected sketch and form the basis for retrieving positions and orientations. Together with the geometry type, this semantics defines the meaning of the sketch, e.g. "Curve–Street".

Gestures are interpreted from their 2D shape without taking into account the underlying scenery. The sketched 2D geometry is analyzed by a shape recognition algorithm, which uses the distance and angle of intermediate gesture points as features for the correlation of drawn gestures and predefined template gestures and results in navigation commands such as "Gesture–CircleLeft".

For composite sketches, the component interpretations are concatenated and thereby represent more complex navigation commands, e.g., "Curve–Street, Point–Building". From these navigation command representations, the navigation system concludes how to generate camera animations from the sketch input.

To improve the usability of our sketch-based navigation, the history of navigation commands and animations is considered. For this, the navigation system stores the past navigation activities (the type of navigation and a possible target identifier). For example, a user points on a building and triggers the navigation "drive to building". If the user points to the same building with the following sketchy navigation command, the system synthesizes a short camera flight around the object allowing the user to "inspect the building".

4.2 Mapping Navigation Commands to Animations

Based on the determined navigation intention, the camera animation is planned. For each supported (composite) sketchy navigation command, the system features a handler comprising the knowledge of how to derive paths and orientations from projected sketches and gestures. For a curve on a street, the curve points are filtered for removing noise and interpreted as a path on that street. Extending this sketch by a point on a building, orients the camera toward the hit surface point. By contrast, a single "point on building" command leads to the computation of the shortest path to that building. The navigation handlers generate camera settings for key frames (e.g., starting point, intermediate points, and end point of a camera path), which are interpolated for creating the animation.

4.3 Visualization of Pending Navigation Commands

As a key element, our extended sketch-based navigation approach incorporates feedback to the users by visual cues about pending navigations. They act as a preview of how the system interprets the sketches, which navigation is determined, and allow the users to verify whether their navigation intention has been correctly recognized.

The navigation cues are integrated into the 3D scene and displayed and animated during the navigation animation. Path arrows on the terrain or street hint at where the camera will move along and billboard-attached target arrows indicate points of interest – see Fig. 3.

4.4 Sketching Speed

The speed at which sketch geometry is drawn can be utilized for improving the sketch-based navigation interface. It can denote the user experience, can be used for determining animation speed, and could influence the animation dramaturgy.

Sketching speed is different for near and far parts of the 3D scene. A path drawn at far distance has a smaller extend on the 2D view plane than a path drawn nearby. Thus, the speed of object-related sketches is calculated from the path speed in 3D. As gestures are not projected into 3D, their speed is calculated from the speed of sketching on the view plane.
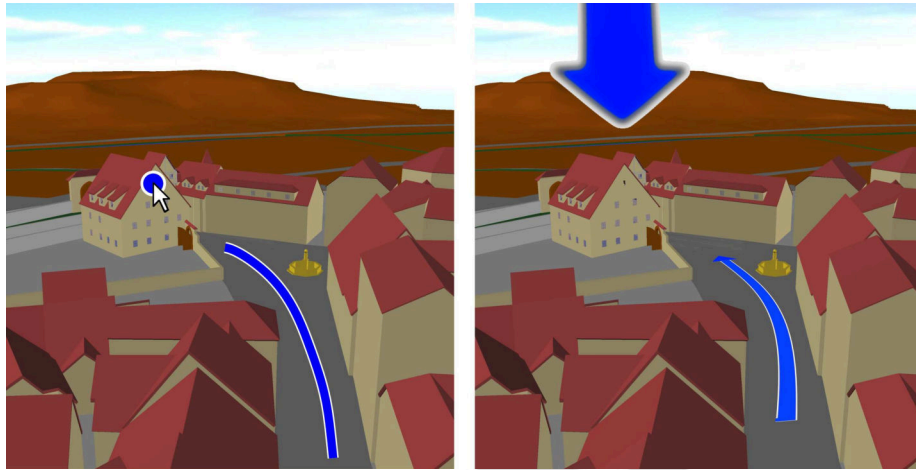
Fig. 3. Example of a sketch-based navigation command (left) and the resulting visual cues integrated in the 3D scene (right). The path arrow symbolizes the camera movement and the target arrow points to the selected building.

5 Conclusions and Future Work

The presented sketch-based navigation commands abstract the navigation process in 3D virtual environments. Instead of controlling and maneuvering the virtual camera, users rather specify their navigation goals, trigger automated navigation processes, and obtain smooth camera animations. The graphical navigation commands are interpreted according to shape, spatial context, and temporal context. The approach takes advantage of the inherent navigation affordances of the scene objects, considers sketching speed, and integrates visual feedback.

Sketch-based navigation lends itself for being used by non-experts on 3D navigation but also for providing task- and user-specific mechanisms to complex navigation operation. Furthermore, sketch-based navigation allows to implement a step-by-step interactive exploration of 3D VEs on thin clients (e.g., mobile devices), which would only handle the sketch input, while the corresponding server manages the 3D VE and renders the camera animations.

In future work, we will investigate task and goal-specific sketchy navigation commands and extend the sketch-based navigation vocabulary including more object types and navigation affordances for virtual 3D city and landscape models. Additionally, we plan to enhance the visual feedback mechanisms to return more information about recognized navigation intentions and to extend the camera dramaturgy taking into account additional 3D object types.

References

1. Henrik Buchholz, Johannes Bohnet, and Jürgen Döllner. Smart and physically-based navigation in 3d geovirtual environments. In *IV '05: Proceedings of the Ninth*

International Conference on Information Visualisation, pages 629–635, Washington, DC, USA, 2005. IEEE Computer Society.

2. Nicholas Burtnyk, Azam Khan, George Fitzmaurice, Ravin Balakrishnan, and Gordon Kurtenbach. Stylecam: Interactive stylized 3d navigation using integrated spatial & temporal controls. In *UIST '02: Proceedings of the 15th annual ACM Symposium on User Interface Software and Technology*, pages 101–110, New York, NY, USA, 2002. ACM.

3. Jonathan M. Cohen, John F. Hughes, and Robert C. Zeleznik. Harold: A world made of drawings. In *NPAR '00: Proceedings of the 1st International Symposium on Non-Photorealistic Animation and Rendering*, pages 83–90, New York, NY, USA, 2000. ACM.

4. Rudolph P. Darken and Barry Peterson. *Handbook of Virtual Environment Technology*, chapter Spatial Orientation, Wayfinding, and Representation, pages 493–518. Lawrence Erlbaum Assoc., New Jersey, 2002.

5. Jürgen Döllner. Constraints as means of controlling usage of geovirtual environments. *Journal of Cartography and Geographic Information Science*, 32(2):69–80, April 2005.

6. Jürgen Döllner, Benjamin Hagedorn, and Steffen Schmidt. An approach towards semantics-based navigation in 3d city models on mobile devices. In Michael P. Peterson Georg Gartner, William Cartwright, editor, *Location Based Services and TeleCartography*, Lecture Notes in Geoinformation and Cartography, pages 357–368, Berlin Heidelberg, 2007. Springer.

7. C. Russo dos Santos, P. Gros, P. Abel, D. Loisel, N. Trichaud, and J. P. Paris. Metaphor-aware 3d navigation. In *INFOVIS '00: Proceedings of the IEEE Symposium on Information Visualization 2000*, pages 155–165, Washington, DC, USA, 2000. IEEE Computer Society.

8. Martin Hachet, Fabrice Decle, Sebastian Knödel, and Pascal Guitton. Navidget for easy 3d camera positioning from 2d inputs. *IEEE Symposium on 3D User Interfaces 2008*, pages 83–89, March 2008.

9. Andrew J. Hanson and Eric A. Wernert. Constrained 3d navigation with 2d controllers. In *VIS '97: Proceedings of the 8th Conference on Visualization '97*, pages 175–182, Los Alamitos, CA, USA, 1997. IEEE Computer Society Press.

10. Takeo Igarashi, Rieko Kadobayashi, Kenji Mase, and Hidehiko Tanaka. Path drawing for 3d walkthrough. In *UIST '98: Proceedings of the 11th annual ACM Symposium on User Interface Software and Technology*, pages 173–174, New York, NY, USA, 1998. ACM.

11. Takeo Igarashi, Satoshi Matsuoka, and Hidehiko Tanaka. Teddy: A sketching interface for 3d freeform design. In *SIGGRAPH '99: Proceedings of the 26th annual Conference on Computer Graphics and Interactive Techniques*, pages 409–416, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co.

12. Azam Khan, Ben Komalo, Jos Stam, George Fitzmaurice, and Gordon Kurtenbach. Hovercam: interactive 3d navigation for proximal object inspection. In *I3D '05: Proceedings of the 2005 Symposium on Interactive 3D Graphics and Games*, pages 73–80, New York, NY, USA, 2005. ACM.

13. Desney S. Tan, George G. Robertson, and Mary Czerwinski. Exploring 3d navigation: Combining speed-coupled flying with orbiting. In *CHI '01: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 418–425, New York, NY, USA, 2001. ACM.

14. Robert C. Zeleznik, Kenneth P. Herndon, and John F. Hughes. Sketch: An interface for sketching 3d scenes. In Holly Rushmeier, editor, *SIGGRAPH '96 Conference Proceedings*, pages 163–170. Addison Wesley, 1996.